AMENDMENTS TO THE SPECIFICATION

Please amend paragraph [0017] as follows:

[0017] As is well known in the art, texture mapping techniques generally utilize a coordinate system for mapping a texture onto a 3D graphical object. For instance, in rendering a texture represented as polygons (e.g., triangles), a texture mapping technique may use a ~~3Dcoordinate~~ 3D coordinate system formed by the surface normal vector, tangent vector, and binormal vector for each vertex. Further, as is well known in the art, texture coordinate gradient vectors are commonly computed for orienting the texture within the coordinate system of the texture mapping technique. Thus, in rendering the texture, a texture mapping technique, such as a bump mapping technique, may compute texture coordinate gradient vectors for each polygon (e.g., triangle) that forms the texture being rendered in order to properly orient the texture within the texture mapping's coordinate system. It should be appreciated that the 3D object onto which the texture is being mapped may comprise many polygons. Because the texture coordinate gradient vectors are computed for each of the polygons, much computation may be associated with the computation of such coordinate gradient vectors. Therefore, it is desirable to provide a method for efficiently computing coordinate gradient vectors in order to optimize the efficiency of the rendering process.

Please amend paragraph [0032] as follows:

[0032] Although PTM textures were primarily developed for three-dimensional texture mapping applications, PTM textures are not limited to three-dimensional applications. For example, PTM textures may be utilized to perform image enhancement. PTM textures may also be utilized to interactively control lighting conditions to promote increased perception of surface characteristics. As one example, PTM may be utilized to provide an efficient method for rendering graphical objects with surface reflectance properties (e.g., PTM may be implemented for modeling a Bidirectional Reflectance Distribution Function), such as disclosed in concurrently filed and commonly assigned U.S. Patent Application Serial ~~No. _____~~, No. 09/921,571, entitled "SYSTEM AND METHOD FOR RENDERING DIGITAL IMAGES HAVING SURFACE REFLECTANCE PROPERTIES." In certain embodiments of the present invention, PTMs may be implemented having four (or more) independent variables, and the PTMs may be parameterized in different ways depending on the independent variables implemented therein. For instance, in certain embodiments PTMs

may include four independent variables with two independent variables used for indexing the PTM and the other two independent variables used to evaluate the PTM function. For example, a PTM may be parameterized using independent variables for representing light direction (e.g., l_u and l_v) and independent variables for representing surface position (e.g., with texture coordinates s and t). The surface position may be used to index the texture, e.g., using texture coordinates (s, t). An example of such a PTM may take the form:

$$PTM(s,t,l_u,l_v) = A(s,t)l_u^2 + B(s,t)l_v^2 + C(s,t)l_u l_v + D(s,t)l_u + E(s,t)l_v + F(s,t).$$

As with the variables u and v described above, l_u and l_v represent scalar quantities associated with orthogonal components of a vector. For example, l_u and l_v may represent the intensity of light from two different directions where the texel is rendered on the three-dimensional object, as described above. And, s and t represent texture coordinates that identify a position on the texture. Independent variables l_u and l_v for representing the light direction may be determined as follows: l_u = Dot(light, tangent) and l_v = Dot(light, binormal). Thus, independent variable l_u is the dot product of the light and tangent vectors, while variable l_v is the dot product of the light and binormal vectors.

Please amend paragraph [0052] as follows:

[0052] In determining such 3D coordinate frame, the computed texture coordinate gradient vectors for a triangle are used to properly orient the 3D coordinate frame for each vertex of the ~~triangle.~~ triangle. In embodiments of the present invention, two gradient vectors are computed for each triangle of a graphical object. One gradient vector ("G_s") is computed which identifies the direction of "maximum change" in the s texture coordinate, and the other gradient vector ("G_t") is computed which identifies the direction of "maximum change" in the t texture coordinate. In this sense, the direction of "maximum change" refers to the direction in which a component of the texture coordinate (i.e., s or t) changes the fastest. For instance, in the example of Fig. 3, gradient vector G_s is determined as being directed along texture coordinate s in the direction from texture coordinate (0,0) to texture coordinate (1,0). Similarly, gradient vector G_t is determined as being directed along texture coordinate t in the direction from texture coordinate (0,0) to texture coordinate (0,1). The determined texture coordinate gradient vectors for the triangle shown in Fig. 4 are illustrated therein.

Please amend paragraph [0055] as follows:

[0055] Because of the change in orientation of the triangle in Fig. 4 from that of Fig. 3, the texture coordinate gradient vectors G_s and G_t are different in Fig. 4. The first effect of such a change is that the calculation of the tangent vector changes for the vertices of the triangle of Fig. 4. It should be noted that the tangent vector for each vertex in Fig. 4 has changed from the tangent vector determined for each ~~vertext~~ vertex in the example of Fig. 3. It should be recognized that the orientation along the *t* texture coordinate has not changed from Fig. 3 to Fig. 4, and therefore the binormal vector remains the same for each vertex. Of course, if the orientation of the triangle in Fig. 4 had a different orientation along the *t* texture coordinate, the binormal vector would change.

Please amend paragraph [0058] as follows:

[0058] As described above, texture coordinate gradient vectors are generally computed for each graphics primitive (e.g., triangle) of a graphical object to be rendered using PTM. For instance, texture coordinate gradient vectors are generally computed for each graphics primitive of a 3D object onto which a texture is being mapped. Because a graphical object may include many such graphics primitives and texture coordinate gradient vectors are computed for each of such graphics primitives, embodiments of the present invention utilize an efficient method for computing the texture coordinate gradient vectors to further improve the rendering efficiency of a PTM. A method for efficiently computing texture coordinate gradient vectors is disclosed in co-pending and commonly assigned Patent Application Serial Number 09/573,059, entitled ~~"METHOD AND APPARATUS FOR PERFORMING H-SPACE LIGHTING IN A GRAPHICS PIPELINE OF A COMPUTER GRAPHICS DISPLAY SYSTEM,"~~ "METHOD AND APPARATUS FOR PERFORMING H-SPACE BUMP MAPPING SUITABLE FOR IMPLEMENTATION WITH H-SPACE LIGHTING IN A GRAPHICS PIPELINE OF A COMPUTER GRAPHICS DISPLAY SYSTEM," from which this application is a continuation-in-part. In accordance with various embodiments of the present invention, such method for efficiently computing texture coordinate gradient vectors is utilized for computing texture coordinate gradient vectors for use with PTMs (as well as other texture mapping techniques in which the resulting texture varies responsive to its orientation relative to a light vector) for rendering graphical objects.

Accordingly, the method for efficiently computing texture coordinate gradient vectors is further reiterated herein below.

Please amend paragraph [0071] as follows:

[0071] Once calculated, P and Q can be used to calculate the gradient vector (A,B,C). In the above equations for computing P and Q, it should be recognized that the denominators in each equation are the same. Also, two gradient vectors will need to be calculated. That is, one gradient vector is calculated using U (i.e., G_s) and one gradient vector is calculated using ~~V (i.e., G_t~~ V(i.e., G_t). The denominators in the calculations of P and Q are independent of U and V and therefore need to be calculated only once. Thus, because the denominators are the same for both P and Q, they only need to be computed once in solving for P and Q. Additionally, because P and Q must be determined for both U and V and because the denominators used in the calculations of P and Q are independent of U and V, once such denominator is computed once, it may be used for both U and V.

Please amend paragraph [0073] as follows:

[0073] Turning now to ~~Fig. 3,~~ Fig. 5, an exemplary computer graphics display system 10 is shown, in which embodiments of the present invention may be implemented. The computer graphics display system 10 comprises a host CPU 12 connected to an input/output (I/O) controller 13, a graphics system 15, a system memory device 16 and a display device 17. The CPU 12 communicates with the system memory 16 and the I/O controller via a system bus 14. The graphics system 15 communicates with the I/O controller 13 via I/O bus 16. A user (not shown) may communicate with the CPU 12 via a peripheral input device, such as a keyboard or mouse, to generate and/or manipulate an image being rendered on the display device 17.

Please amend paragraph [0074] as follows:

[0074] An example of components that may be included in graphics system 15 of the computer graphics display system 10 in accordance with embodiments of the present invention are further shown in ~~Fig. 4.~~ Fig. 6. Graphics system 15 includes buffer ~~412~~ 612, graphics processor ~~414,~~ 614, parametric texture map ~~416,~~ 616, frame buffer ~~418,~~ 618, and display ~~420.~~ 620. Buffer ~~412~~ 612 holds geometry data that describes a 3D object which is to be generated on the display ~~420.~~ 620. Buffer ~~412~~ 612 may be any suitable data storage

mechanism now known or later discovered, including as examples Random Access Memory (RAM), cache memory, disk drive, floppy disk, and optical disc. The 3D object is represented in the buffer ~~412~~ 612 as a set of polygons in a 3D space. In one embodiment, the polygons are triangles and the geometry data in buffer ~~412~~ 612 includes the 3D coordinates of the vertices of the triangles.

Please amend paragraph [0075] as follows:

[0075] Graphics system 15 includes graphics processor ~~414~~ 614, which may be any suitable processor now known or later discovered, including without limitation any processor from the Itanium™ family of processors or a PA-8500 processor available from Hewlett-Packard Company. Graphics processor ~~414~~ 614 reads the parameters that define the polygons from buffer ~~412~~ 612 and scan converts each polygon. The scan conversion of a polygon yields a 2D view of the polygon which depends on a view direction and light source direction. A 2D view of a polygon includes a color value for each pixel of the polygon which is visible in the plane of display ~~420~~ 620. Graphics processor ~~414~~ 614 writes the color values for the rendered polygons into frame buffer ~~418~~ 618. Frame buffer ~~414~~ 614 may be any suitable data storage mechanism, including as examples RAM, cache memory, and disk drive. The color values from frame buffer ~~418~~ 618 may be provided to display ~~420~~ 620 on a frame by frame basis. Display ~~420~~ 620 may be any conventional 2D display device now known or later discovered, such as a scan device or flat-panel display device, as examples.

Please amend paragraph [0076] as follows:

[0076] Parametric texture map ~~416~~ 616 may hold parameters that define a surface structure in a manner in which the appearance of the surface structure varies with any user-defined vector, such as the view vector, the light source vector, or the half-angle vector. In general, the half-angle vector is a vector that is halfway between the view and light source vectors. Graphics processor ~~414~~ 614 maps the surface structure defined in parametric texture map ~~416~~ 616 onto the polygons obtained from buffer ~~412~~ 612 during scan conversion. The result is a more realistic rendering of 3D features in a surface on a 3D object in comparison to many other texture mapping techniques.